

Fork and build (Windows)

Requirements

- Register with GitHub and obtained a profile. This allows the setting up of repositories where your code can be published.
- [Visual Studio 2013](#)
- [Git](#)
- [CMake](#)

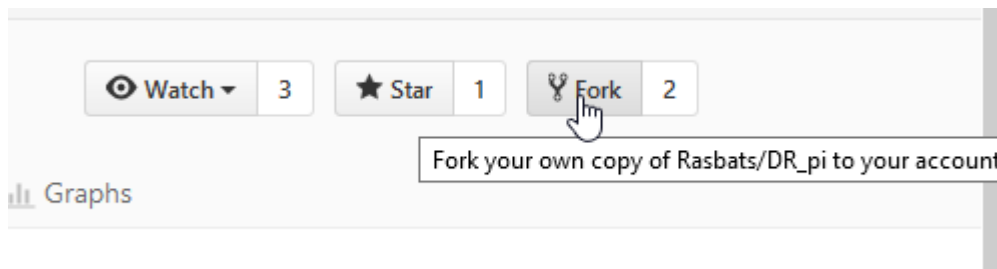
Process

Obtain a copy of DR_pi

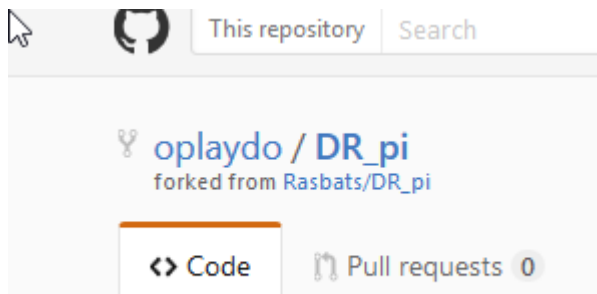
Register with GitHub and login to your profile.

Navigate to https://github.com/Rasbats/DR_pi

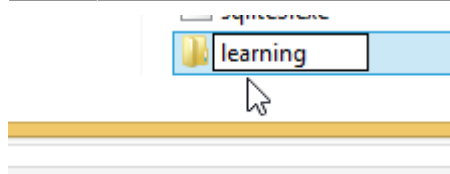
Press the 'Fork' button



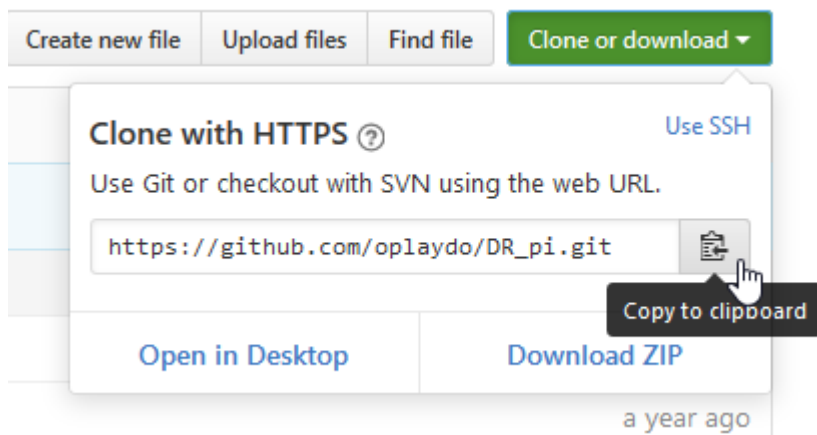
Go back to your own profile and you will find a find a copy of DR_pi



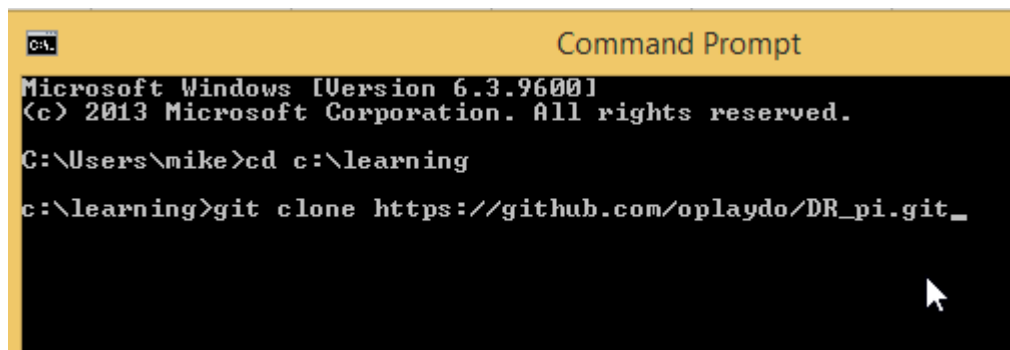
Make a folder for your projects. Here it is 'C:\learning'



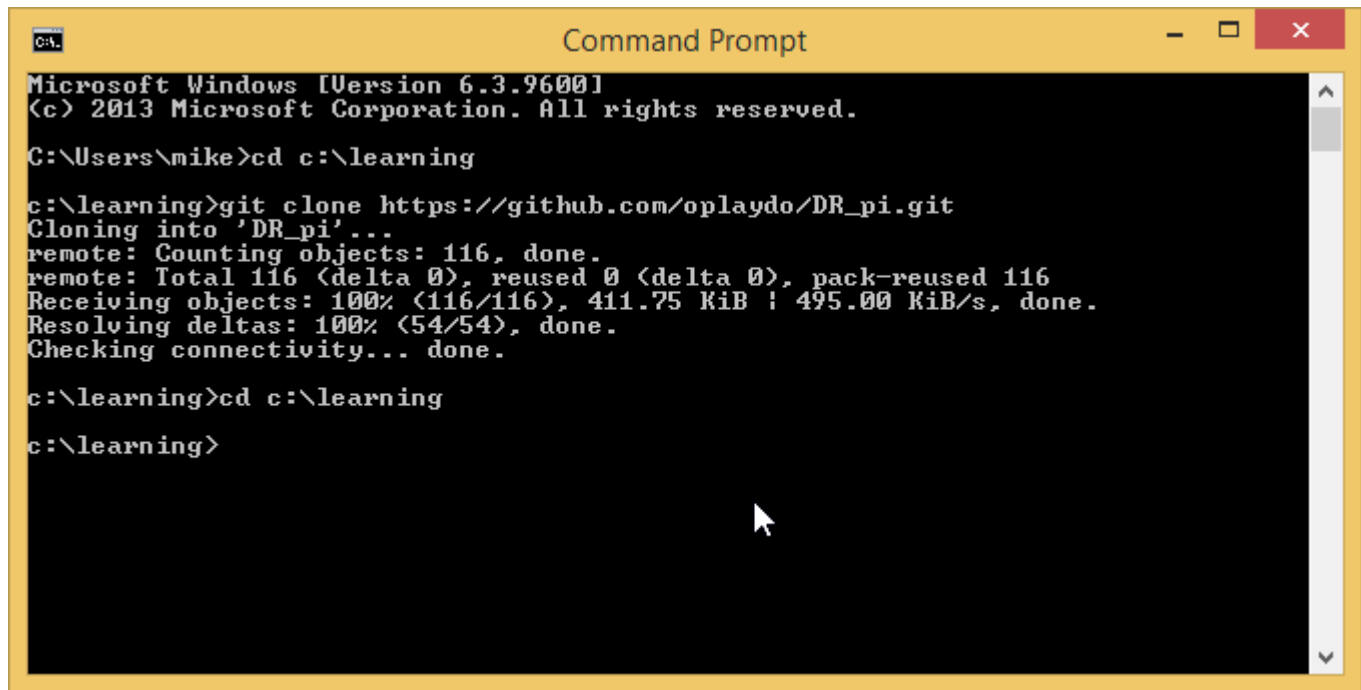
Go back to your repository and select 'Clone or download'. Using 'Copy' to add the link to your clipboard.



Open a command prompt, cd to 'C:\learning'. Type 'git clone ' and then right-click paste.



Run this.



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\mike>cd c:\learning

c:\learning>git clone https://github.com/oplaydo/DR_pi.git
Cloning into 'DR_pi'...
remote: Counting objects: 116, done.
remote: Total 116 (delta 0), reused 0 (delta 0), pack-reused 116
Receiving objects: 100% (116/116), 411.75 KiB | 495.00 KiB/s, done.
Resolving deltas: 100% (54/54), done.
Checking connectivity... done.

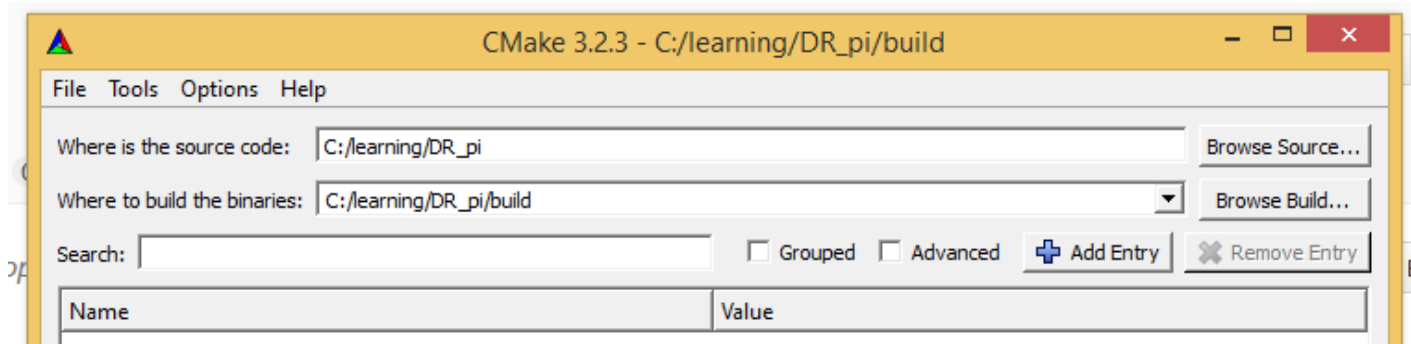
c:\learning>cd c:\learning

c:\learning>
```

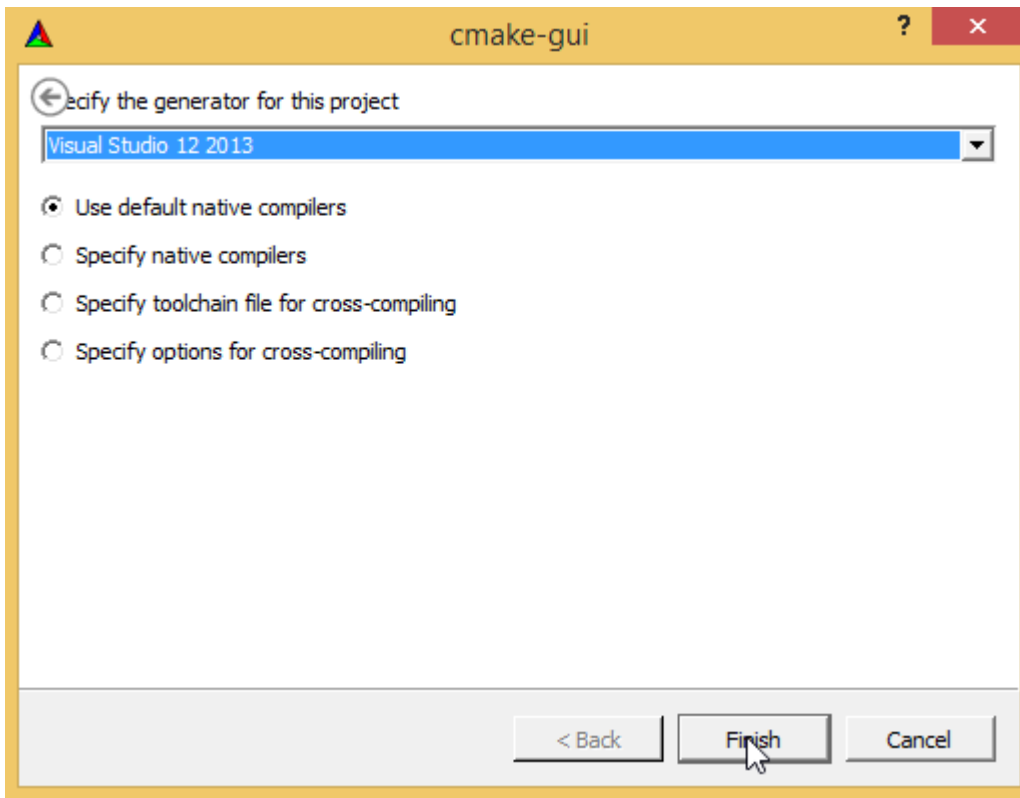
You will find a new folder has been made under learning called 'DR_pi'.

Using CMake

Because I have two versions of wxWidgets on my machine I prefer to use CMake-gui.

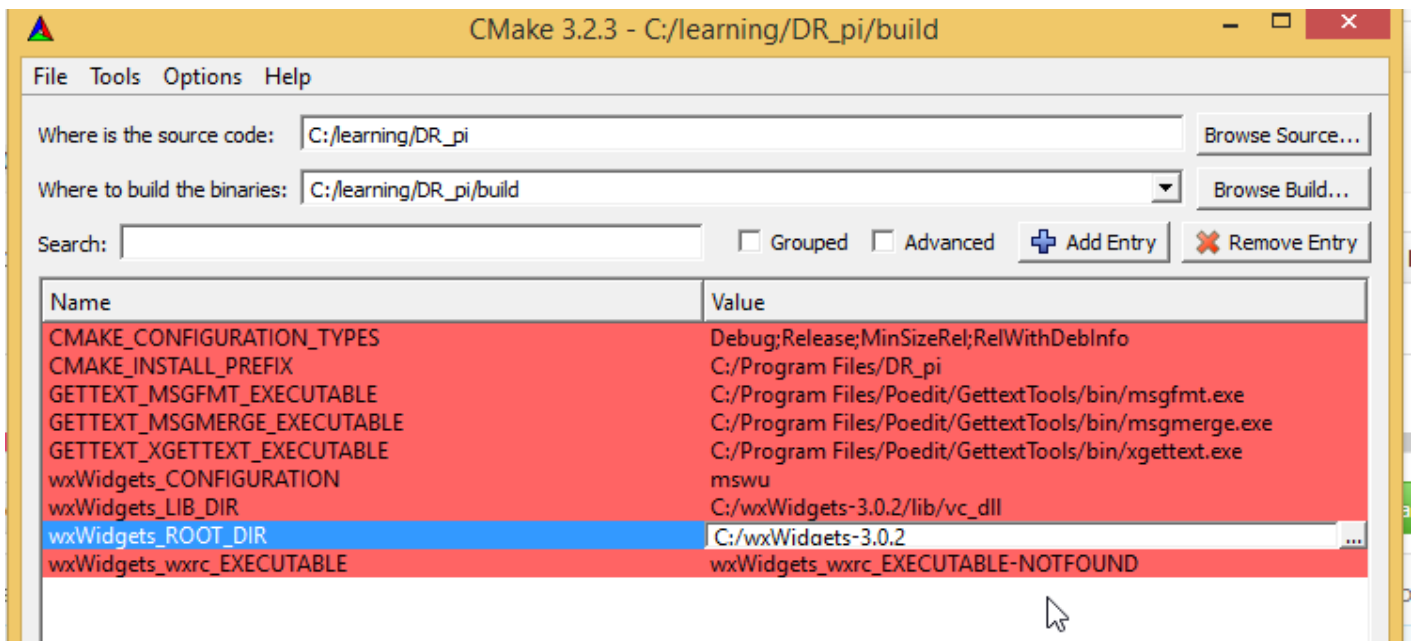


The source code folder is entered and where the build will be made. If this does not exist it will be created for you.



And press the 'Configure' button.

On this next screen a few changes need to be made.



The 'msw' has been changed to 'mswu' to use unicode. The 'vc_dll' folder of wxWidgets is pointing to the correct version of wxWidgets, and the main wxWidgets folder is correct. We need to be using wxWidgets 3.0.2.

Now press 'Generate' to make the VS2013 solution in the 'build' folder.

Local Disk (C:) > learning > DR_pi > build >				Search build
Name	Date modified	Type	Size	
CMakeFiles	06/02/2017 10:55	File folder		
ALL_BUILD.vcxproj	06/02/2017 10:55	VCXPROJ File	22 KB	
ALL_BUILD.vcxproj.filters	06/02/2017 10:55	VC++ Project Filte...	1 KB	
cmake_install.cmake	06/02/2017 10:55	CMAKE File	8 KB	
CMakeCache.txt	06/02/2017 10:55	Text Document	25 KB	
CPackConfig.cmake	06/02/2017 10:53	CMAKE File	4 KB	
CPackSourceConfig.cmake	06/02/2017 10:53	CMAKE File	4 KB	
DR_pi.sln	06/02/2017 10:55	Microsoft Visual S...	7 KB	
DR_pi.vcxproj	06/02/2017 10:55	VCXPROJ File	38 KB	
DR_pi.vcxproj.filters	06/02/2017 10:55	VC++ Project Filte...	3 KB	

Using Visual Studio 2013

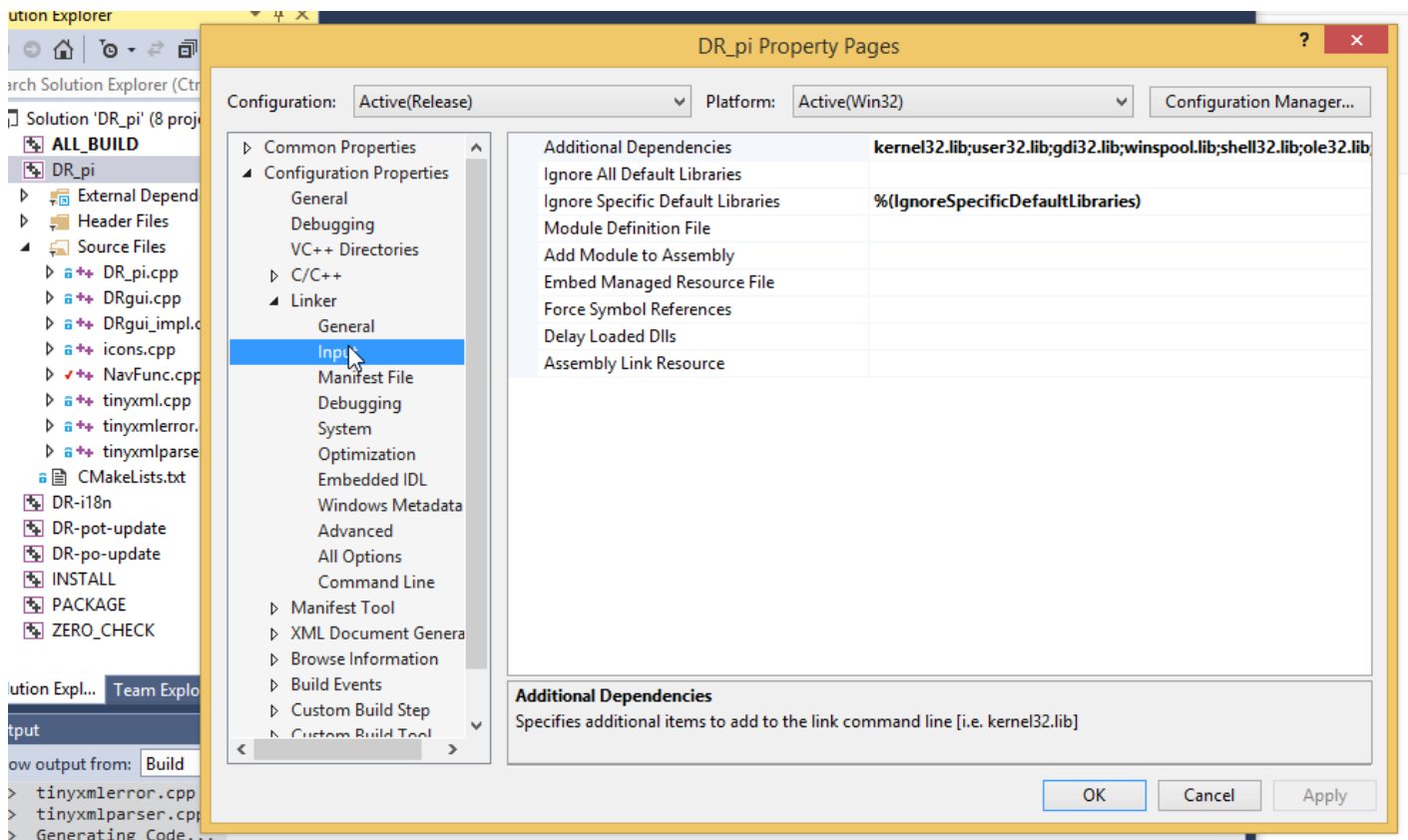
To allow standalone compilation you need to download 'opencpn.lib' from [here](#) (unzip and extract the file). Place it in the same 'build' folder shown above.

Your build folder will look like this:

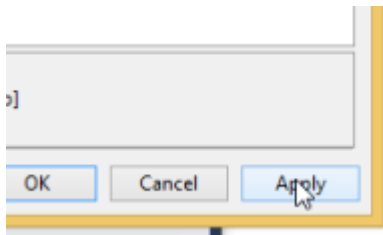
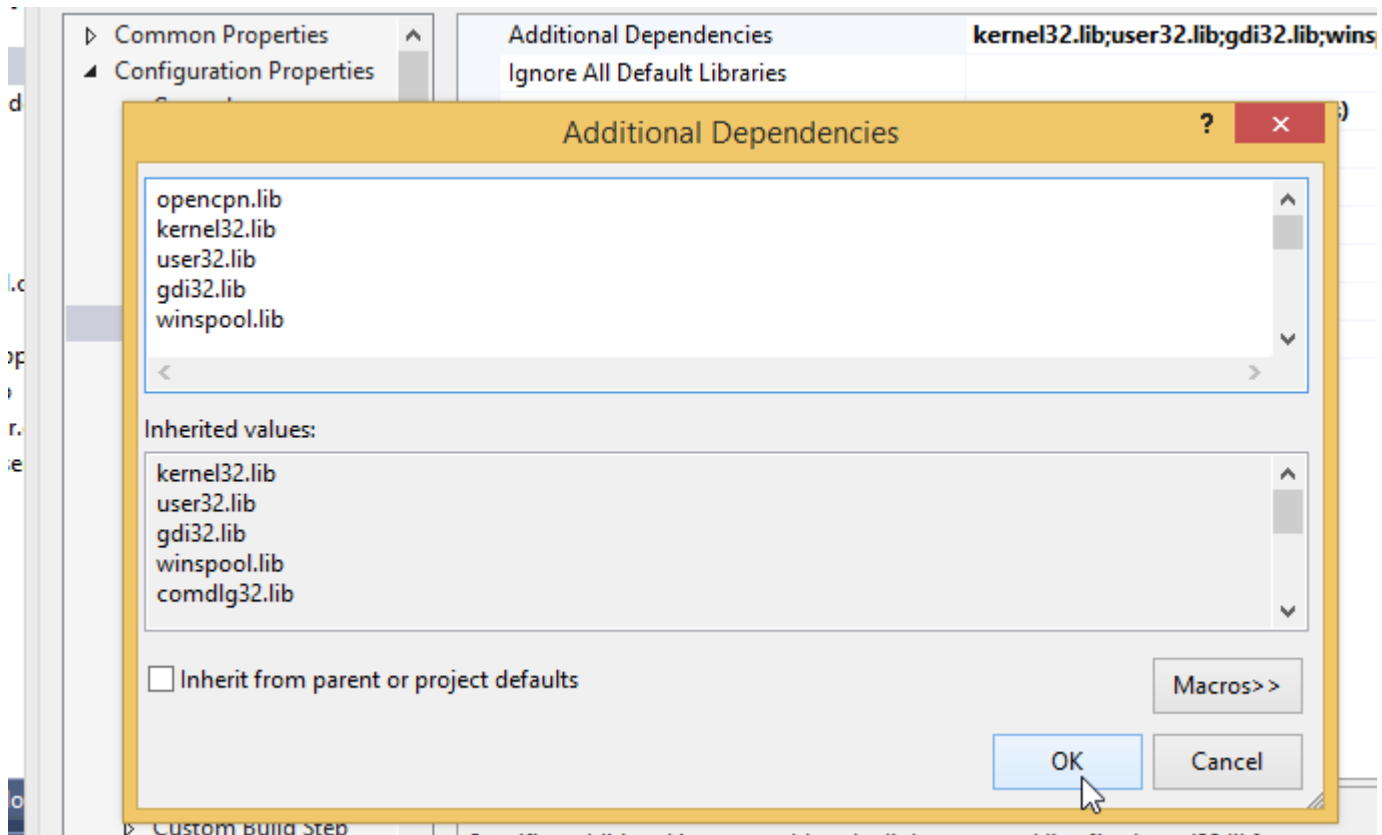
DR_pi.sdf	27/01/2017 20:45	SQL Server Comp...	45,652 KB
DR_pi.sln	30/10/2015 13:02	Microsoft Visual S...	7 KB
DR_pi.v12.suo	27/01/2017 20:45	Visual Studio Solu...	47 KB
DR_pi.vcxproj	21/01/2017 17:44	VCXPROJ File	73 KB
DR_pi.vcxproj.filters	21/01/2017 17:44	VC++ Project Filte...	2 KB
DR-i18n.vcxproj	30/10/2015 13:02	VCXPROJ File	56 KB
DR-i18n.vcxproj.filters	30/10/2015 13:02	VC++ Project Filte...	1 KB
DR-pot-update.vcxproj	30/10/2015 13:02	VCXPROJ File	64 KB
DR-pot-update.vcxproj.filters	30/10/2015 13:02	VC++ Project Filte...	1 KB
DR-po-update.vcxproj	30/10/2015 13:02	VCXPROJ File	59 KB
DR-po-update.vcxproj.filters	30/10/2015 13:02	VC++ Project Filte...	1 KB
INSTALL.vcxproj	30/10/2015 13:02	VCXPROJ File	12 KB
INSTALL.vcxproj.filters	30/10/2015 13:02	VC++ Project Filte...	1 KB
opencpn.lib	18/11/2015 12:41	LIB File	121 KB
PACKAGE.vcxproj	30/10/2015 13:02	VCXPROJ File	12 KB
PACKAGE.vcxproj.filters	30/10/2015 13:02	VC++ Project Filte...	1 KB
ZERO_CHECK.vcxproj	30/10/2015 13:02	VCXPROJ File	39 KB

Open the solution file (.sln) with Visual Studio 2013.

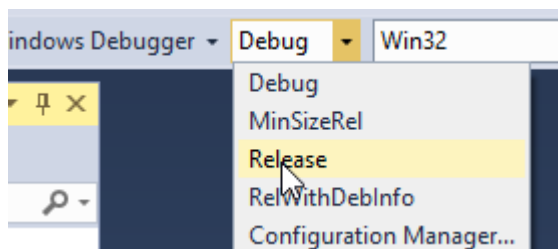
First step is to link the opencpn.lib library you have just downloaded. Right-click on 'DR_pi' in 'Solution Explorer'. This opens a 'Property' page.



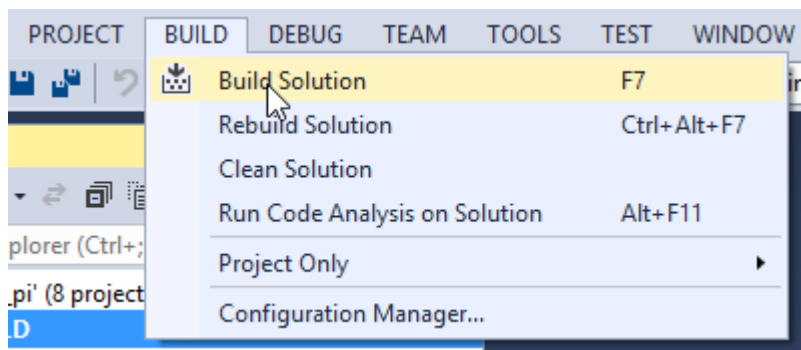
Select 'Linker', 'Input'. On the right you see 'Additional Dependencies'. This line can be edited by using a drop-down. Make a new line and add 'opencpn.lib' to the top of the list. Then 'OK' and 'Apply' what you have changed.



Now back in the main screen choose 'Release' from the drop-down on the toolbar.



And the final step is to press 'Build'. Use 'Build Solution' or 'F7' for building and compiling the plugin.

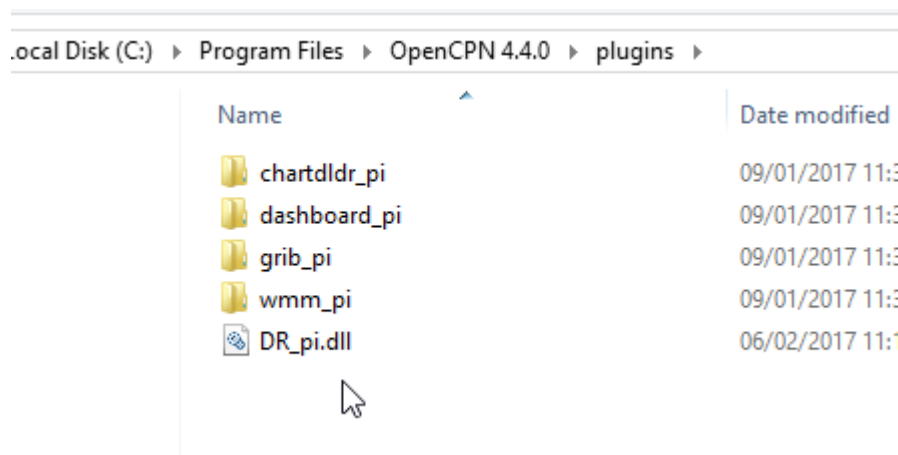


The plugin is built in 'learning\DR_pi\build\Release' folder. It is called "DR_pi.dll".

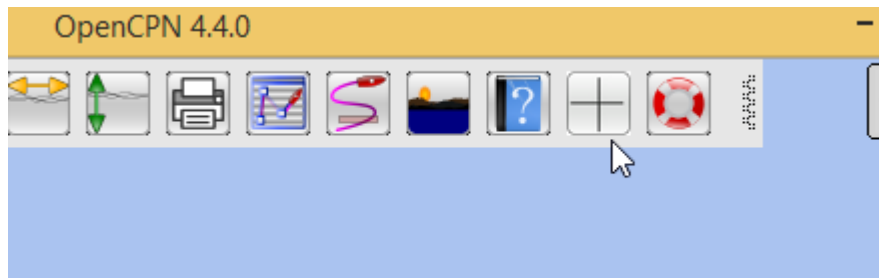
If you get an error saying 'Fabs' could not be found, try replacing 'Fabs' with 'abs'.

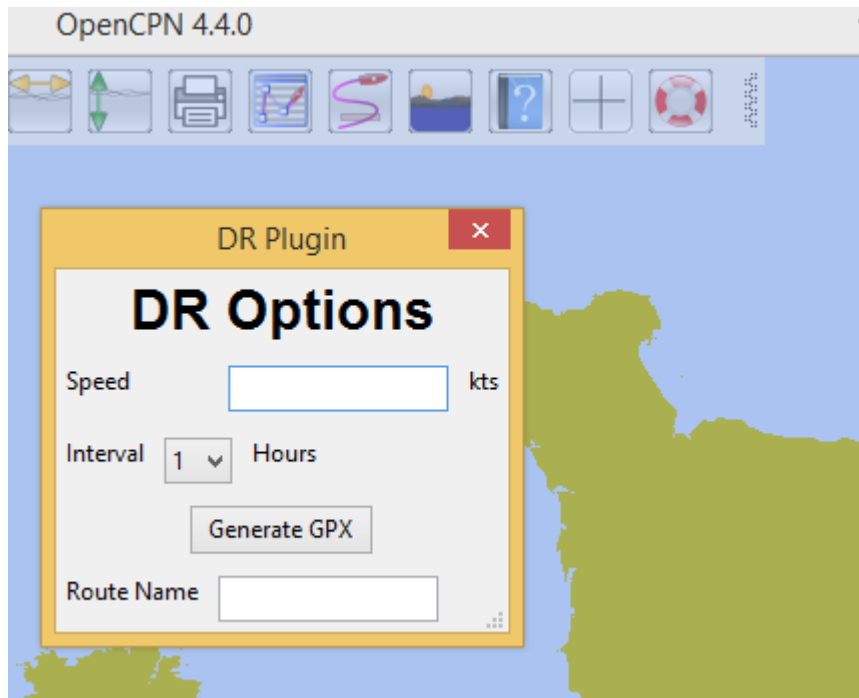
Installing the plugin

Copy 'DR_pi.dll' from the 'Release' folder to the plugins folder of the main OpenCPN program.



When you enable the DR plugin you should see the plugin icon on the toolbar.





Notes

The copy of 'opencpn.lib' downloaded to use with this guide was made with OpenCPN 4.2.0 source code. The plugin will still run in O 4.4.0 because plugins need to be upwardly compatible.

When functions from new versions of the API are needed you will need to source a new version of 'opencpn.lib'.

The next guide will show the DR_pi plugin being renamed and modified.

'oplaydo1_pi' will allow the user to input start and finish positions. From these positions a GPX file will be created that can be imported and viewed in OpenCPN.

From:
<https://opencpn.org/wiki/dokuwiki/> - OpenCPN Manuals

Permanent link:
https://opencpn.org/wiki/dokuwiki/doku.php?id=opencpn:developer_manual:plugins:learning:fork_build_windows

Last update: 2017/03/01 11:20

